Advanced Java Lab

Week-11

Roll Number: 238W1A12C4

1. Set up H2 in-memory DB with Spring Data JPA and create an entity

Code:

Dependencies:

Spring web, spring data jpa, h2 database

Student.java:

```
package com.example.demo;


import jakarta.persistence.Entity;
import jakarta.persistence.Id;


@Entity
public class Student {


    @Id
    private int id;
    private String name;


    public Student() {}


    public Student(int id, String name) {
        this.id = id;
        this.name = name;
    }


    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
}
```

application.properties:

```
spring.h2.console.enabled=true
spring.h2.console.path=/h2-console

spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=

spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
```

2. Create Repository and Demonstrate CRUD

StudentRepository.java:

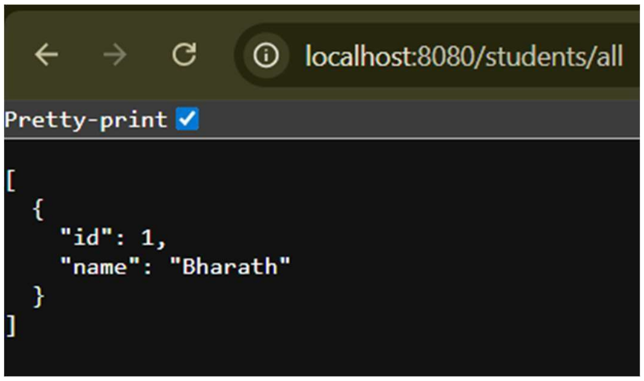
```
package com.example.demo;

import org.springframework.data.jpa.repository.JpaRepository;

public interface StudentRepository extends JpaRepository<Student, Integer> {
}
```

StudentController.java:

```
package com.example.demo;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;

@RestController
@RequestMapping("/students")
public class StudentController {

    @Autowired
    private StudentRepository repo;

    // CREATE
    @PostMapping("/add")
    public Student add(@RequestBody Student s) {
        return repo.save(s);
    }

    // READ all
    @GetMapping("/all")
    public List<Student> getAll() {
        return repo.findAll();
    }

    // READ by ID
    @GetMapping("/{id}")
    public Student getById(@PathVariable int id) {
        return repo.findById(id).orElse(null);
    }
    // UPDATE
    @PutMapping("/update")
    public Student update(@RequestBody Student s) {
        return repo.save(s);
    }

    // DELETE
    @DeleteMapping("/delete/{id}")
    public String delete(@PathVariable int id) {
        repo.deleteById(id);
        return "Deleted student with id: " + id;
    }
}
```

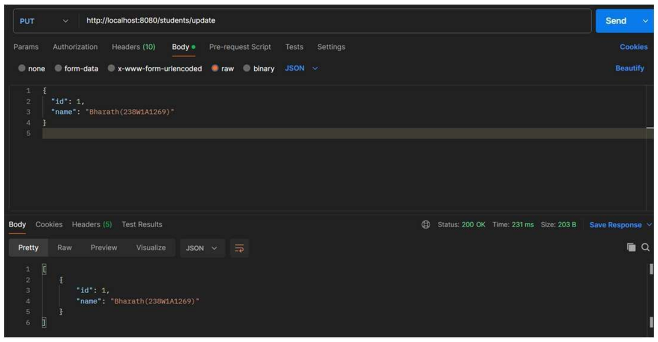
Output:



localhost:8080/students/all

Pretty-print ✓

```
[
  {
    "id": 1,
    "name": "Bharath"
  }
]
```



PUT http://localhost:8080/students/update

none form-data x-www-form-urlencoded [✓ raw] binary JSON

```
{
  "id": 1,
  "name": "Bharath(238W1A1269)"
}
```

Status: 200 OK Time: 231 ms Size: 203 B

```
[
    {
        "id": 1,
        "name": "Bharath(238W1A1269)"
    }
]
```